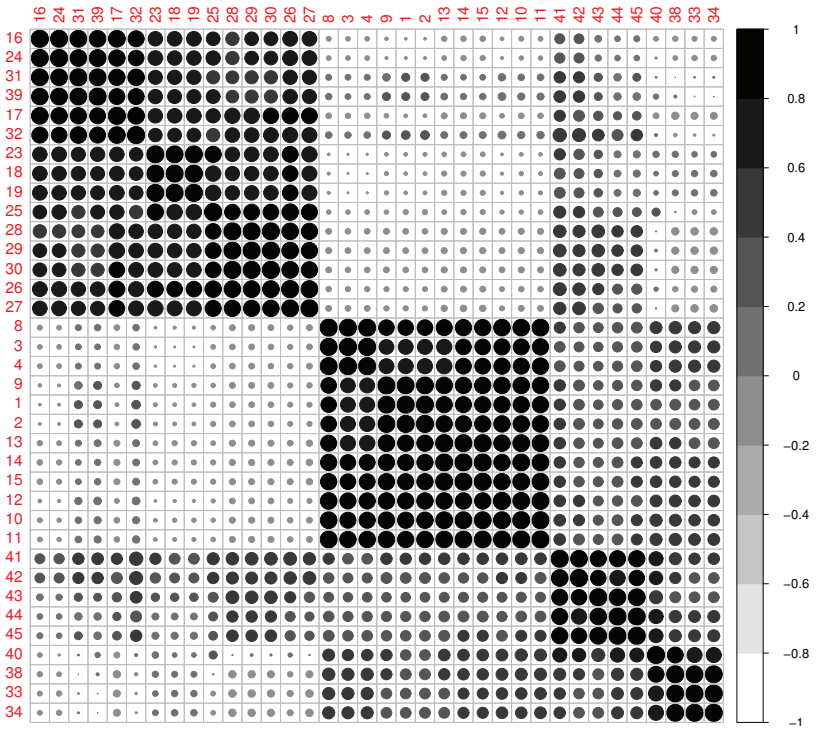

Figure 2. Corrélation de Spearman pour les caractéristiques du jeu de données OHSUMED

moins marqué, même s'il reste important : nos approches suppriment 2 à 3 fois plus de caractéristiques que les méthodes de référence.

Les algorithmes de pondération sont donc beaucoup plus efficaces que l'état de l'art pour effectuer la sélection de variables. En effet, elles obtiennent des qualités d'ordonnancement similaires, tout en supprimant jusqu'à 10 fois plus de caractéristiques. En ce sens, elles sont bien plus performantes que les approches de sélection de variables existantes en discrimination et en apprentissage d'ordonnancement.

5.1.2. Temps d'exécution et ratios de parcimonie

Les figures 5 et 6 présentent conjointement les temps d'exécution et les ratios de parcimonie obtenus pour les algorithmes que nous proposons et l'état de l'art. Notons que nous ne fournissons pas les valeurs pour FenchelRank, qui est le seul algorithme à ne pas être implémenté en Matlab.

Nous observons que les temps d'exécution des algorithmes de pondération que nous proposons sont globalement stables aux seuils de 50 % et 30 %, mais qu'ils augmentent lorsque le seuil de 10 % est considéré. Ce comportement pourrait traduire le

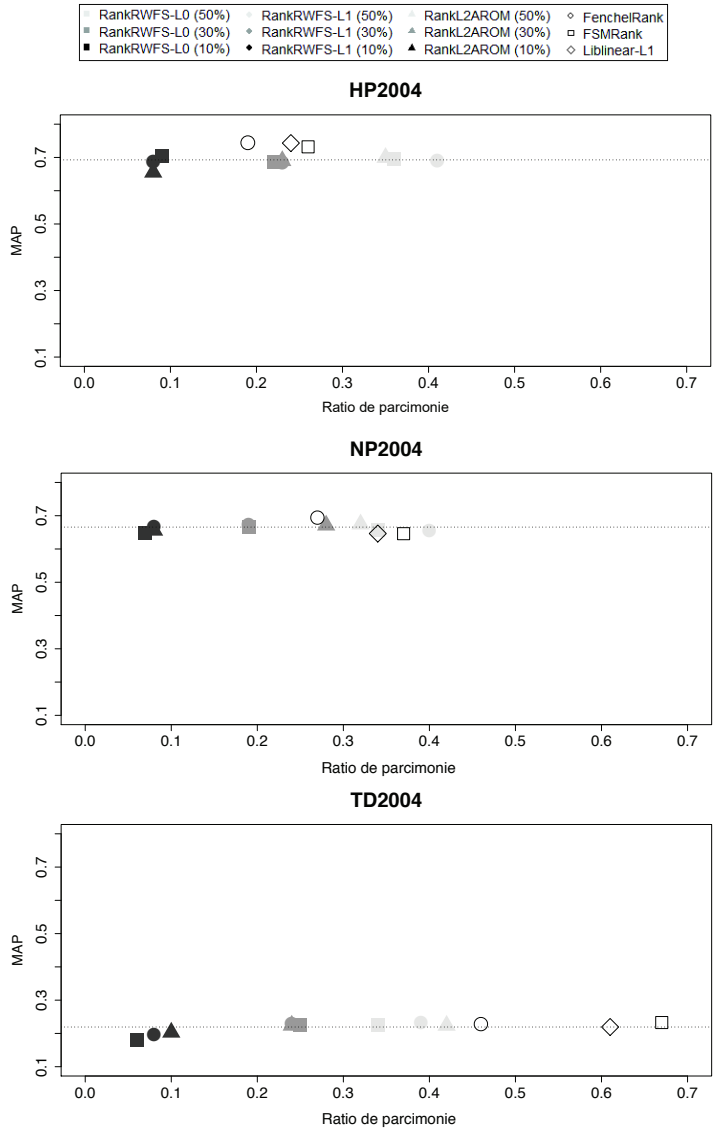

Figure 3. MAP vs. ratios de parcimonie pour chaque algorithme sur les trois jeux de données de référence issus de la collection GOV de LETOR 3.0. La ligne en pointillé représente la valeur de MAP moyenne pour l'ensemble des méthodes. Les algorithmes situés à gauche sont les plus parcimonieux. Les algorithmes que nous proposons sont équivalents en matière de MAP et meilleurs en matière de parcimonie

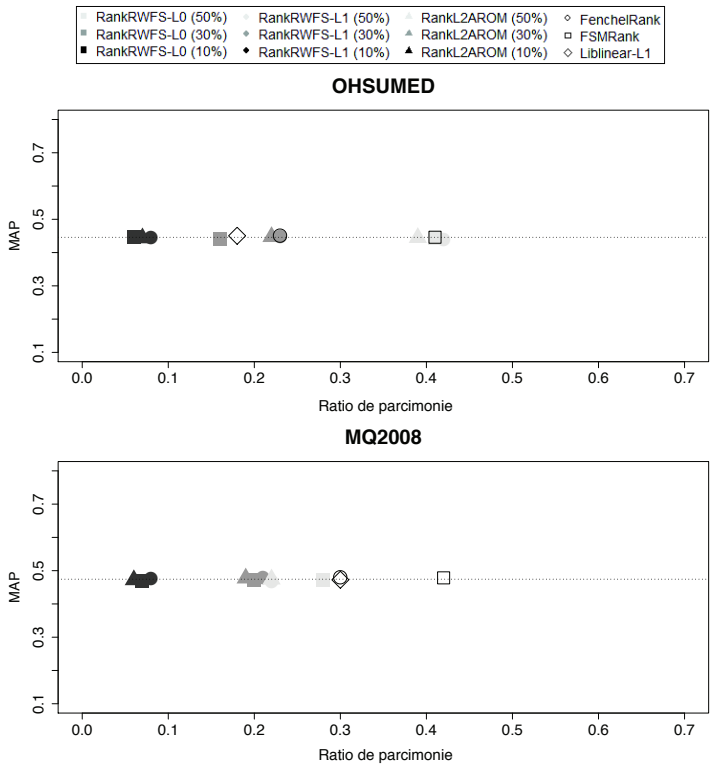

Figure 4. MAP vs. ratios de parcimonie pour chaque algorithme sur les jeux de données de référence OHSUMED et MQ2008. La ligne en pointillé représente la valeur de MAP moyenne pour l'ensemble des méthodes. Les algorithmes situés à gauche sont les plus parcimonieux. Les algorithmes que nous proposons sont équivalents en matière de MAP et meilleurs en matière de parcimonie

fait que la sélection devient plus difficile quand le nombre de caractéristiques restantes devient très faible.

Nous remarquons que les algorithmes de pondération que nous proposons sont plus performants que les méthodes de l'état de l'art. En effet, lorsque nous considérons les plus petits seuils de parcimonie, nos approches sont soit plus parcimonieuses, soit à la fois plus parcimonieuses et plus rapides.

Nous nous plaçons au seuil de 10 %. Sur le jeu de données TD2004, Rankℓ_2-AROM, RankRWFS-ℓ_1 et RankRWFS-ℓ_0 sont respectivement 2.5, 6 et 7 fois plus rapides que Liblinear-ℓ_1, tandis que RankRWFS-ℓ_1 et RankRWFS-ℓ_0 sont respectivement 2 et 3 fois plus rapides que FSMRank. Sur Ohsumed, les méthodes que nous proposons sont en moyenne 4 fois plus rapides que FSMRank et 2 fois plus rapides que Liblinear-ℓ_1. Sur HP2004, NP2004 et MQ2008, nos algorithmes sont plus lents que FSMRank, mais plus rapides que Liblinear-ℓ_1, tout en sélectionnant moins de ca-

ractéristiques que ces derniers. Plus précisément, sur MQ2008, nos algorithmes sont 2 à 4 fois plus rapides que Liblinear-ℓ_1. Ils sont 2 fois plus lents que FSMRank dans le pire des cas, mais sélectionnent environ 75 % de caractéristiques en moins. Sur HP2004, nos algorithmes sont tous 2 fois plus rapides que Liblinear-ℓ_1, 4 à 5 fois plus lents que FSMRank, tous en supprimant 2 à 3 fois plus de caractéristiques. Sur NP2004, RankRWFS-ℓ_0 et Rankℓ_2-AROM respectivement 2 et 3 fois plus rapides que Liblinear-ℓ_1. RankRWFS-ℓ_1 et Rankℓ_2-AROM sont 2 à 3 fois plus lents que FSMRank, mais suppriment 4 à 5 fois plus de caractéristiques. Parmi les algorithmes de pondération que nous proposons, RankRWFS-ℓ_0 est le plus rapide sur tous les jeux de données.

Les méthodes de sélection par pondération de la norme ℓ_2 que nous proposons sont donc plus performantes que les approches de l'état de l'art en apprentissage d'ordonnancement et en discrimination, puisqu'elles conservent une qualité d'ordonnancement similaire, tout en étant plus parcimonieuses et plus rapides. Par ailleurs, RankRWFS-ℓ_0 est globalement la plus rapide des trois approches proposées, ainsi que la plus parcimonieuse. Cela justifie d'une part, l'utilisation d'une approche de pondération de la norme ℓ_0 différente des approches de type ℓ_2-AROM et, d'autre part, l'utilisation de la norme ℓ_0.

5.1.3. Caractéristiques sélectionnées

Dans cette étude, nous nous sommes intéressés aux caractéristiques sélectionnées par les modèles, dans le but d'extraire un ensemble de critères importants pour l'ordonnancement. Nous considérons les modèles appris au seuil de 10 %.

Les algorithmes de pondération sélectionnent globalement les mêmes caractéristiques pour un même jeu de données. Ils se comportent donc de façon cohérente. Sur MQ2008, les algorithmes ont sélectionné les caractéristiques 23 et 39 pour l'ensemble des répétitions ainsi que les critères 32 et 37 pour la majorité des modèles. Toutes ces caractéristiques correspondent aux scores obtenus via des modèles de langue, connus en RI pour être très performants. Sur Ohsumed, les caractéristiques 3, 4 (basées sur la fréquence des termes), 11, 41 (score BM25 sur le titre et le document complet) et 43 (score modèle de langue) sont les plus fréquemment sélectionnées. Sur la collection Gov, les caractéristiques sélectionnées varient légèrement suivant le jeu de données. Sur HP2004, les algorithmes sélectionnent majoritairement les caractéristiques 23 (score BM25 du titre), 46 (hyperlink based feature propagation) et 52 (HostRank). Sur NP2004, les caractéristiques 23 et 46 sont également sélectionnées, ainsi que la caractéristique 22 (score BM25 de l'ancre). Par contre, la caractéristique 52 n'est généralement pas conservée. Enfin, sur TD2004, les algorithmes sélectionnent une combinaison des caractéristiques 22, 23, 52 et ponctuellement 46 suivant la répétition considérée. Toutes ces mesures sont connues pour être hautement informatives.

Les algorithmes proposés dans ces travaux sont donc capables d'apprendre des fonctions d'ordonnancement facilement interprétables, cohérentes et de bonne qualité.

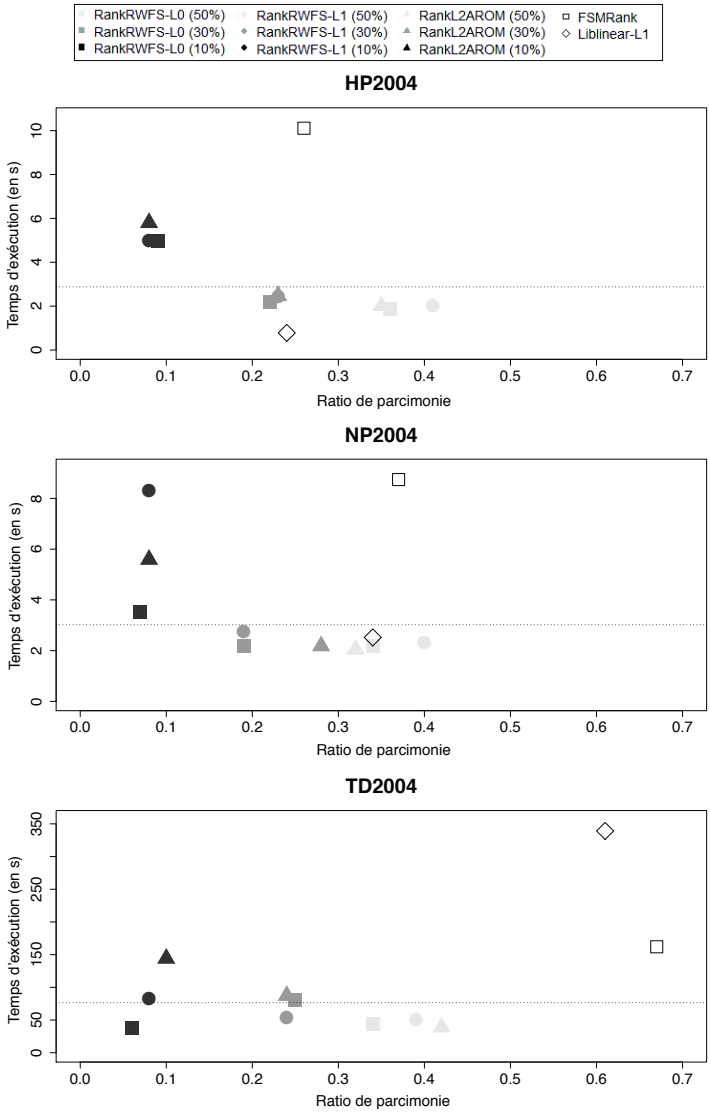

Figure 5. Temps d'exécution vs. ratios de parcimonie pour chaque algorithme sur les trois jeux de données de référence issus de la collection Gov de LETOR 3.0. La ligne pointillée représente le temps d'exécution moyen pour l'ensemble des méthodes. Les algorithmes situés à gauche et en bas sont les plus parcimonieux et les plus rapides. Les algorithmes que nous proposons sont meilleurs à la fois en matière de parcimonie et de rapidité d'exécution

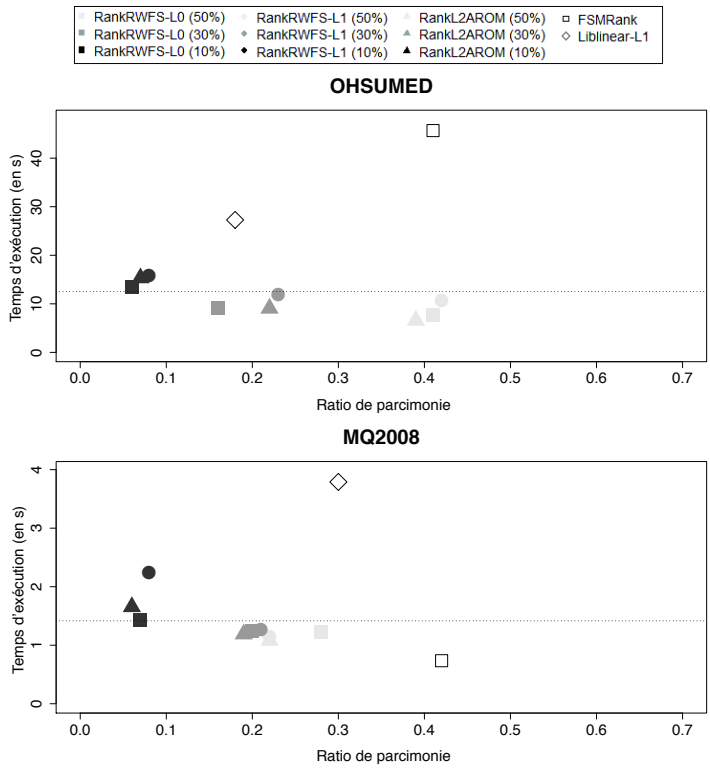

Figure 6. Temps d'exécution vs. ratios de parcimonie pour chaque algorithme sur les jeux de données OHSUMED et MQ2008. La ligne pointillée représente le temps d'exécution moyen pour l'ensemble des méthodes. Les algorithmes situés à gauche et en bas sont les plus parcimonieux et les plus rapides. Les algorithmes que nous proposons sont meilleurs en matière de parcimonie et de rapidité d'exécution

5.2. Expérimentations sur un jeu de données à plus large échelle

Dans cette section, nous évaluons nos algorithmes sur un jeu de données plus volumineux, dans un contexte d'application industrielle sur un moteur de recherche commercial. Nous nous intéressons à la qualité d'ordonnancement, aux ratios de parcimonie et aux temps d'exécution. Nous nous sommes concentrés ici sur l'évaluation et la comparaison des quatre algorithmes RankRWFS-ℓ_1, RankRWFSℓ_0, Rankℓ_2-AROM et FSMRank.

La figure 7 présente le lien entre MAP et ratio de parcimonie sur le jeu de données industriel à plus large échelle. La ligne en pointillé représente la valeur moyenne de MAP obtenue par les différentes méthodes.

Nous observons que les algorithmes présentent des qualités d'ordonnancement globalement similaires. L'algorithme le moins "bon" dégrade la MAP de 0,16 % com-

parativement à FSMRank, tandis que le "meilleur" l'améliore de 1 %. Les valeurs de MAP restent donc stables pour tous les algorithmes et tous les seuils de sélection.

De la même façon que sur les jeux de données LETOR, nous constatons que nos méthodes permettent de supprimer plus de caractéristiques que l'état de l'art, tout en conservant la même qualité d'ordonnancement. Ainsi, les algorithmes RankRWFS-ℓ_1, RankRWFS-ℓ_0 et Rankℓ_2-AROM obtiennent des ratios de parcimonie au seuil de 30 % similaire à l'algorithme de référence FSMRank ($0,26$, $0,2$ et $0,18$ contre $0,22$). Ils sont capables de supprimer de 3 à 4 fois plus de caractéristiques que FSMRank au seuil de 10 %. Ils sont ainsi beaucoup plus efficaces que l'algorithme de l'état de l'art pour apprendre des fonctions d'ordonnancement parcimonieuses.

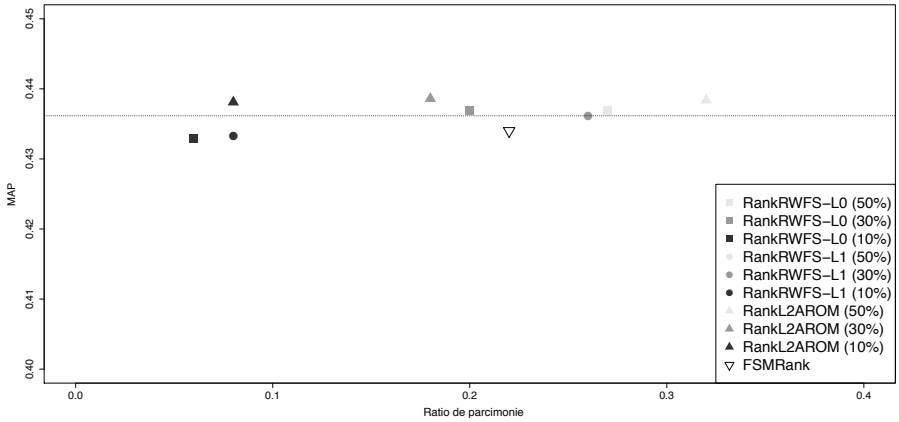

Figure 7. MAP vs. ratios de parcimonie pour chaque algorithme sur le jeu de données NOMAO. La ligne en pointillé représente la valeur de MAP moyenne pour l'ensemble des méthodes. Les algorithmes situés à gauche sont les plus parcimonieux. Les algorithmes que nous proposons sont équivalents en matière de MAP et meilleurs en matière de parcimonie

La figure 8 présente conjointement les temps d'exécution et les ratios de parcimonie obtenus pour les algorithmes que nous proposons et FSMRank, méthode de référence. La ligne en pointillé représente le temps d'exécution moyen, exprimé en secondes, pour les différentes méthodes. Les algorithmes les plus parcimonieux et/ou les plus rapides sont situés à gauche et en bas de la figure.

Nous observons que l'algorithme le plus rapide est l'algorithme de référence FSMRank, avec une durée d'exécution d'environ 20 secondes. Notons que 75 % du temps d'exécution de cet algorithme est à imputer au calcul de la matrice des corrélations entre caractéristiques qui est utilisée pour effectuer la sélection.

Parmi les algorithmes que nous proposons, l'algorithme RankRWFS-ℓ_1 est globalement le plus rapide, avec des temps d'exécution compris entre 20 et 35 secondes suivant le seuil de sélection considéré. Il semble également être le plus stable. En effet, les temps d'exécution aux seuils de 10 % et 30 % sont globalement similaires (30 et

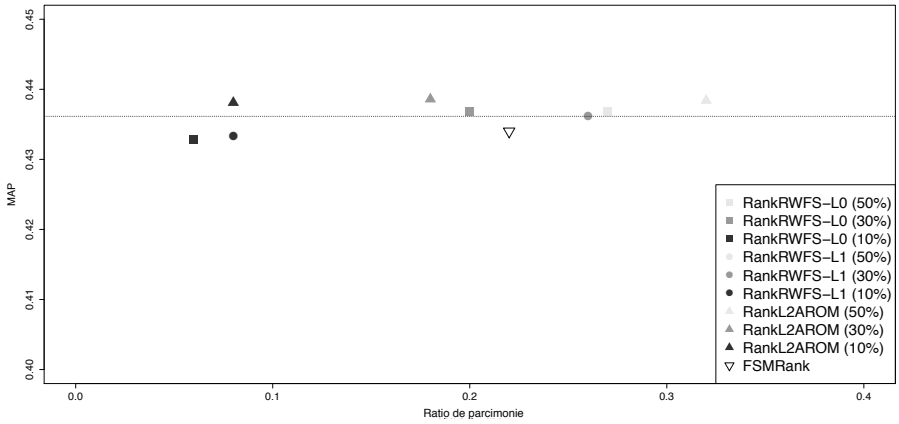

Figure 8. Temps d'exécution vs. ratios de parcimonie pour chaque algorithme sur le jeu de données Nomao. La ligne en pointillé représente le temps d'exécution moyen pour l'ensemble des méthodes. Les algorithmes situés à gauche et en bas sont les plus parcimonieux et les plus rapides. Les algorithmes que nous proposons sont plus lents que l'état de l'art, mais plus parcimonieux

35 secondes) tandis que ceux de RankRWFS-ℓ_0 et Rankℓ_2-AROM sont multipliés par deux entre le seuil à 30 % et le seuil à 10 %. Ce comportement avait déjà été observé sur les jeux de données LETOR.

Nous constatons que les algorithmes que nous proposons permettent de supprimer de 3 à 4 fois plus de caractéristiques que FSMRank, en prenant de $1,5$ à $3,5$ fois plus de temps. Le rapport temps d'exécution - nombre de caractéristiques supprimées serait ainsi plus favorable pour les algorithmes que nous proposons.

En conclusion, les algorithmes que nous proposons utilisent de 5 à 8 caractéristiques dans les fonctions d'ordonnancement, contre 22 en moyenne pour l'algorithme de référence, et 101 initialement, sans dégrader la qualité d'ordonnancement et dans des temps d'exécution qui restent très raisonnables.

6. Conclusion

Dans cet article, nous nous sommes intéressés à l'adaptation et à l'analyse des méthodes de pondération des SVM en norme ℓ_2 pour la sélection de variables via des SVM parcimonieux en norme ℓ_1 et ℓ_0. A notre connaissance, il s'agit des premiers travaux à proposer l'utilisation de la norme ℓ_0 et des méthodes de pondération de la norme ℓ_2 en sélection de variables pour l'apprentissage d'ordonnancement.

Nos expérimentations ont montré que :

1. Les méthodes de pondération de la norme ℓ_2 que nous proposons sont plus efficaces que l'état de l'art pour supprimer un grand nombre de caractéristiques des fonc-

tions d'ordonnancement. Elles conservent une qualité d'ordonnancement équivalente aux méthodes de référence, tout en utilisant de 3 à 10 fois moins de caractéristiques.

2. Les méthodes proposées obtiennent des qualités d'ordonnancement comparables à l'état de l'art, tout en étant de 2 à 7 fois plus rapides.

3. Les algorithmes que nous proposons permettent de sélectionner un sous-ensemble de caractéristiques très informatives pour chaque jeu de données. Ces sous-ensembles sont cohérents.

4. Les algorithmes que nous proposons permettent de supprimer plus de 90 % des caractéristiques sur l'ensemble des jeux de données considérés. Ce chiffre n'est pas nécessairement étonnant vu la grande redondance constatée dans les jeux de données et présentée dans cet article.

Les méthodes de pondération de la norme ℓ_2 que nous proposons constituent des approches performantes en sélection de variables pour l'apprentissage d'ordonnancement. Les performances obtenues concernant les ratios de parcimonie sont importantes, car la réduction du nombre de caractéristiques, en plus de simplifier les modèles appris, permet de gagner du temps lors de l'extraction des valeurs des caractéristiques. Les trois algorithmes Rank-ℓ_2-AROM, RankRWFS-ℓ_1 et RankRWFS-ℓ_0 présentent des performances globalement similaires, bien que RankRWFS-ℓ_0 semble plus rapide et légèrement plus parcimonieux. Par ailleurs, l'approche générique RankRWFS est plus flexible, puisqu'elle permet d'utiliser différentes régularisations parcimonieuses par une modification de la règle de mise à jour. Notons également que n'importe quel algorithme de résolution des SVM en norme ℓ_2 peut être incorporé au sein de ces méthodes. D'autres ré-écritures des normes (Chartrand, Yin, 2008 ; Zhang, Kingsbury, 2010) pourraient également être utilisées dans le cadre de cette approche. Leur apport fera l'objet de travaux futurs. Nous prévoyons également d'intégrer de nouvelles ré-écritures pour approcher la norme ℓ_0 par pondération de SVM en norme ℓ_1 et de comparer la performances de notre approche à des méthodes considérant d'autres régularisations parcimonieuses (Laporte, Flamary *et al*., 2014).

Remerciements

Les auteurs remercient l'entreprise Nomao ainsi que la Région Midi-Pyrénées, qui ont contribué au financement de ces travaux (financement 10009018).

Bibliographie

Burges C., Shaked T., Renshaw E., Lazier A., Deeds M., Hamilton N. *et al*. (2005). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on machine learning*, p. 89–96.

Cao Z., Qin T., Liu T.-Y., Tsai M.-F., Li H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on machine learning*, p. 129–136.

Chapelle O., Keerthi S. S. (2010). Efficient algorithms for ranking with svms. *Information Retrieval*, vol. 13, n° 3, p. 201–215.

Chartrand R., Yin W. (2008). Iteratively reweighted algorithms for compressive sensing. In *Icassp*, p. 3869-3872.

Dang V., Croft B. (2010). Feature selection for document ranking using best first search and coordinate ascent. In *Sigir workshop on feature generation and selection for information retrieval.*

Fan R.-E., Chang K.-W., Hsieh C.-J., Wang X.-R., Lin C.-J. (2008). Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, vol. 9, p. 1871–1874.

Freund Y., Iyer R., Schapire R. E., Singer Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, vol. 4, p. 933–969.

Geng X., Liu T.-Y., Qin T., Li H. (2007). Feature selection for ranking. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval*, p. 407–414.

Harrell F. E. (2001). *Regression modeling strategies: With applications to linear models, logistic regression, and survival analysis*. Springer.

Harrell F. E. J., Dupont C., many others. (2015). Hmisc: Harrell miscellaneous Manuel de logiciel. Consulté sur http://CRAN.R-project.org/package=Hmisc (R package version 3.15-0)

Hastie T., Tibshirani R., Friedman J. H. (2003). *The Elements of Statistical Learning* (Corrected éd.). Springer. Hardcover.

Hua G., Zhang M., Liu Y., Ma S., Ru L. (2010). Hierarchical feature selection for ranking. In *Proceedings of the 19th international conference on world wide web*, p. 1113–1114.

Joachims T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, p. 133-142.

Kira K., Rendell L. A. (1992). A practical approach to feature selection. In *Proceedings of the ninth international workshop on machine learning*, p. 249–256.

Kohavi R., John G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, vol. 97, n° 1-2, p. 273–324.

Kujala J., Aho T., Elomaa T. (2009). A walk from 2-norm svm to 1-norm svm. In *Proceedings of the 2009 ninth ieee international conference on data mining*, p. 836–841.

Lai H., Pan Y., Liu C., Lin L., Wu J. (2013). Sparse learning-to-rank via an efficient primal-dual algorithm. *IEEE Transaction on Computers*, vol. 62, n° 6, p. 1221–1233.

Lai H., Pan Y., Yong T., Yong R. (2013). Fsmrank: A feature selection algorithme for learning-to-rank. *IEEE Transactions on Neural Networks and Learning Systems*.

Lai H., Tang Y., Luo H.-X., Pan Y. (2011). Greedy feature selection for ranking. In *Proceedings of the 15th international conference on computer supported cooperative work in design*, p. 42-46.

Laporte L. (2013). *La sélection de variables en apprentissage d'ordonnancement pour la recherche d'information : vers une approche contextuelle.* Thèse de doctorat, Université Paul Sabatier, Toulouse, France.

Laporte L., Déjean S., Mothe J. (2013). Multiple Clicks Model for Web Search of Multi-clickable Documents (short paper). In *International Conference on Enterprise Information Systems (ICEIS), Angers, 04/07/13-07/07/13*, p. (electronic medium). http://www.scitepress.org/.

Laporte L., Déjean S., Mothe J. (2014). Séparateurs à Vaste Marge pondérés en norme L2 pour la sélection de variables en apprentissage d'ordonnancement (regular paper). In *Conférence francophone en Recherche d'Information et Applications (CORIA), Nancy, 19/03/2014-21/03/2014*, p. 295–310.

Laporte L., Flamary R., Canu S., Déjean S., Mothe J. (2014). Non-convex Regularizations for Feature Selection in Ranking with Sparse SVM. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, n° 6, p. 1118–1130.

Liu T.-Y. (2011). *Learning to rank for information retrieval*. Springer.

Pahikkala T., Airola A., Naula P., Salakoski T. (2010). Greedy rankrls: a linear time algorithm for learning sparse ranking models. In *Sigir 2010 workshop on feature generation and selection for information retrieval*, p. 11-18.

Pan F., Converse T., Ahn D., Salvetti F., Donato G. (2009). Feature selection for ranking using boosted trees. In *Proceedings of the 18th ACM conference on information and knowledge management*, p. 2025–2028.

Sun Z., Qin T., Tao Q., Wang J. (2009). Robust sparse rank learning for non-smooth ranking measures. In *Proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval*, p. 259–266.

Weston J., Elisseeff A., Schölkopf B., Tipping M. (2003). Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, vol. 3, p. 1439-1461.

Xu J., Li H. (2007). Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval*, p. 391–398.

Yu H., Oh J., Han W.-S. (2009). Efficient feature weighting methods for ranking. In *Proceedings of the 18th ACM conference on information and knowledge management*, p. 1157–1166.

Zhang Y., Kingsbury N. (2010). Fast l0-based sparse signal recovery. In *Machine learning for signal processing (mlsp), 2010 ieee international workshop on*, p. 403-408.